When FastText Pays Attention: Efficient Estimation of Word Representations using Constrained Positional Weighting

Vít Novotný[*] and **Michal Štefánik** and **Eniafe Festus Ayetiran** and **Petr Sojka**
Faculty of Informatics
Masaryk University
Brno, Czech Republic
{witiko,stefanik.m,ayetiran}@mail.muni.cz, sojka@fi.muni.cz

Abstract

Since the seminal work of Mikolov et al. (2013a) and Bojanowski et al. (2017), word representations of shallow log-bilinear language models have found their way into many NLP applications. Mikolov et al. (2018) introduced a positional log-bilinear language model, which has characteristics of an attention-based language model and which has reached state-of-the-art performance on the intrinsic word analogy task. However, the positional model has never been evaluated on qualitative criteria or extrinsic tasks and its speed is impractical.

We outline the similarities between the attention mechanism and the positional model, and we propose a *constrained positional model*, which adapts the sparse attention mechanism of Dai et al. (2019). We evaluate the positional and constrained positional models on three novel qualitative criteria and on the extrinsic language modeling task of Botha and Blunsom (2014).

We show that the positional and constrained positional models contain interpretable information about word order and outperform the subword model of Bojanowski et al. (2017) on language modeling. We also show that the constrained positional model outperforms the positional model on language modeling and is twice as fast.

“Words do not mean, people do.” (Wright, 2008)

1 Introduction

Word representations of shallow log-bilinear language models (LBLs) have found many applications in natural language processing (NLP), such as word similarity, word analogy, and language modeling (Bojanowski et al., 2017) as well as word sense disambiguation (Chen et al., 2014), text classification (Kusner et al., 2015), semantic text similarity (Charlet and Damnati, 2017), and information retrieval (Novotný et al., 2020, Section 4). Recently, Devlin et al. (2018) have introduced the deep attention-based language model BERT, which has redefined the state of the art for eleven NLP tasks and turned LBLs to a baseline. Independently, Mikolov et al. (2018) have introduced a positional LBL, which resembles attention-based language models and which has reached state-of-the-art performance on the intrinsic word analogy task.

Later, Clark et al. (2019) have shown that ensembling LBLs with BERT improves performance on the dependency parsing task compared to either LBLs or BERT alone, which has reinvigorated the fading interest in LBLs. Although surprising, the results of Clark et al. (2019) are supported by cognitive psychology: Kahneman (2011) describes the human mind as an interplay of the fast, intuitive, and emotional System 1, and the slow, effortful and logical System 2. Peters et al. (2006) have shown that systems 1 and 2 are mutually supportive and that System 1 adapts to new and more challenging tasks by coaching System 2 to take over its more menial tasks. If we treat Kahneman’s systems 1 and 2 as a metaphor for LBLS and BERT, the results of Clark et al. (2019) seems natural.

In our paper, we describe the relationship between the attention mechanism and the positional LBL of Mikolov et al. (2018), and we propose our constrained positional LBL that adapts the attention sparsification techniques of Dai et al. (2019); Child et al. (2019); Beltagy et al. (2020); Zaheer et al. (2020) for LBLs. We also develop three novel qualitative criteria, which we use to evaluate the positional and constrained positional LBLs in addition to the extrinsic language modeling task.

The rest of our paper is structured as follows: In Section 2, we describe the dense and sparse at-

[*] The first author’s work was graciously funded by the South Moravian Centre for International Mobility as a part of the Brno Ph.D. Talent project.

tention mechanisms, and we relate them to the positional LBL of Mikolov et al. (2018) and our proposed constrained positional LBL. In Section 3, we describe our experimental setup and propose three novel qualitative evaluation measures. In Section 4, we discuss the results of our experiments. We conclude in Section 5 by summarizing our results and suggesting directions for future work.

2 Models

In this section, we describe the dense attention mechanism and we relate it to the positional LBL. Additionally, we describe attention sparsification techniques and we use them to develop our proposed constrained positional LBL.

2.1 Attention

In this section, we describe the purpose of attention in neural machine translation (NMT), and we describe sparsification techniques that make attention computationally tractable.

Dense attention Early neural machine translation used *encoder-decoder models*, where an encoder recurrent neural network (RNN) would first read and encode a *source sequence* into a *fixed-length context vector* and a decoder RNN would then produce a *translated target sequence* from the context vector (Sutskever et al., 2014, Cho et al. (2014b)). Due to the context vector's fixed length, translation performance would deteriorate for longer sequences (Cho et al., 2014a). To enable the translation of longer source sequences, Bahdanau et al. (2016) equipped the decoder with an *attention* mechanism. Instead of having a single encoded vector for the entire source sequence, the attention would construct a different context vector for each target word. Here, the context vector would be a weighted average of the encoder's hidden states, where the weights would be trained to relate relevant source words to the target word.

Following the success of attention in NMT, Cheng et al. (2016) proposed to use the attention mechanism directly in the long-short-term memory (LSTM) cells of RNNs: Instead of computing the current *memory* and *hidden state* using the previous memory and hidden state alone, the current memory and hidden state would be computed as weighted averages of all previous memories and hidden states. Attention would act as a random-access memory mechanism, enabling the LSTM to recall long-range memories. Later, Vaswani et al. (2017) proposed the *Transformer* architecture, which successfully replaced recurrence by the vertical stacking of attention.

Sparse attention Since the attention mechanism learns weights for all pairs of source and target words, its space complexity is quadratic in the source sequence length. Several *attention sparsification* schemes have been proposed in literature to enable the translation of longer source sequences by making the space complexity linear (Dai et al., 2019; Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020). Although attention enables the recollection of long-range memories, sparse attention makes it computationally tractable to do so.

2.2 Log-bilinear language models

In this section, we propose our constrained positional model that learns word representations while taking into account morphology and the mutual positions of words. When modeling positions of words, we only use a sparse subset of word vector features, following our observation that only a fraction of a word's meaning depends on its position in a sentence, whereas the rest of its meaning is either fixed or depends on a broader context.

We first present the general word2vec model of Mikolov et al. (2013a) and Mikolov et al. (2013b), followed by the subword fastText model of Bojanowski et al. (2017), and the positional model of Mikolov et al. (2018). Finally, we propose our constrained positional model together with its theoretical foundations, computational benefits, and its close relation to the sparse attention mechanism described in Section 2.1.

General model Mikolov et al. (2013a) introduced the continuous bag of words (CBOW) model, which learns word representations by predicting a masked word w_t from its context $C_t = w_{t-c}, \ldots, w_{t-1}, w_{t+1}, \ldots, w_{t+c}$, where c is *window size* and $w_1, \ldots, w_T$ is the *training corpus*:

$$\arg\min_{\boldsymbol{\theta}} \left[L(\boldsymbol{\theta}) = -\sum_{t=1}^{T} \log \Pr(w_t \mid C_t; \boldsymbol{\theta}) \right]. \quad (1)$$

To estimate $\Pr(w_t \mid C_t)$, Mikolov et al. (2013b) used a simplified variant of the noise contrastive approximation (Gutmann and Hyvärinen, 2012), which they called *negative sampling*:

$$\Pr(w_t \mid C_t) = \sigma(s(w_t, C_t)) \prod_{n \in N_{C_t}} \sigma(-s(n, C_t)), \quad (2)$$

where σ is the logistic function $x \mapsto {}^{1}\!/_{1+e^{-x}}$, N_{C_t} is a set of negative examples n for context C_t, and $s(w_t, C_t)$ is a scoring function that measures how well the masked word w_t matches the context C_t:

$$s(w_t, C_t) = \boldsymbol{u}_{C_t}^{\mathsf{T}} \cdot \boldsymbol{v}_{w_t}, \boldsymbol{u}_{C_t} = \frac{1}{|C_t|}\sum_{w \in C_t} \boldsymbol{u}_w. \tag{3}$$

Here, $\boldsymbol{u}_w \in \mathbb{R}^D$ is the *input vector* of the context word w, $\boldsymbol{v}_{w_t} \in \mathbb{R}^D$ is the *output vector* of the masked word w_t, $\boldsymbol{u}_{C_t}$ is the *context vector* and D is the number of word vector features, which is usually in the low hundreds.

Li (1992) has shown that if we order words $w_{(i)}$ by their decreasing relative frequencies $f_{w_{(i)}}$ in a corpus, then $f_{w_{(i)}}$ exhibits a power law:

$$f_{w_{(i)}} = \frac{c}{i^{\alpha}}, \text{ where } c \approx 0.1 \text{ and } \alpha \approx 1. \tag{4}$$

This law, originally proposed for English by Zipf (1932), shows that most words in our training corpus will only represent a small subset of our vocabulary. By the end of the training, CBOW will have overfit the input and output vectors of the few most frequent words, whereas it will have underfit the input and output vectors of most other words.

To equalize the number of training samples for vocabulary words, Mikolov et al. (2013b) discard corpus words with the following probability:

$$\Pr_{\text{discard}}(w_t) = \max\left(0, 1 - \sqrt{\frac{r}{f_{w_t}}}\right), \tag{5}$$

where the low-pass threshold r ensures that rare words w_t with $f_{w_t} \leq r$ are never discarded.

Subword model The CBOW model only learns representations for words that are present in the training corpus. Additionally, vectors for different inflectional forms of a word share no weights, which delays training convergence for morphologically rich languages.

In response, Bojanowski et al. (2017) have extended CBOW by modeling subwords instead of words: The input vector $\boldsymbol{u}_w$ for a word w become a sum of the input vectors $\boldsymbol{u}_g$ for the subwords $g \in G_w$ of w:

$$\boldsymbol{u}_w = \sum_{g \in G_w} \boldsymbol{u}_g. \tag{6}$$

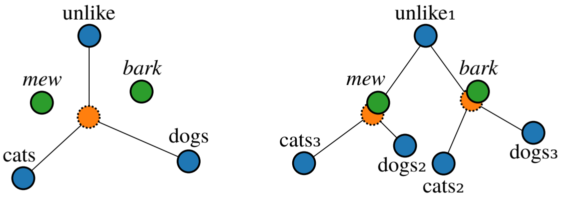

Figure 1: The input (blue), context (orange), and output (green) vectors for the sentences “Unlike dogs, cats ⟨*masked word*⟩.” and “Unlike cats, dogs ⟨*masked word*⟩.” with two masked words *mew* and *bark* in the subword (left) and positional (right) models.

Positional model In many sentences, the position of context words influences their syntactic function, which is important for predicting the masked word. Consider the following two sentences, which produce an identical context vector $\boldsymbol{u}_{C_t}$ despite the different masked words:

1. Unlike dogs, cats ⟨*masked word*⟩.
2. Unlike cats, dogs ⟨*masked word*⟩.

If the context C_t is large, distant context words will only introduce noise to the context vector $\boldsymbol{u}_{C_t}$.

To better adapt to these situations, we would like to have separate input vectors $\boldsymbol{u}_{w,p}$ for different positions $p \in P$ of a context word w:

$$\boldsymbol{u}_{C_t} = \frac{1}{|P|}\sum_{p \in P} \boldsymbol{u}_{w_{t+p},p}. \tag{7}$$

See also Figure 1. Since this would increase the size of the vocabulary by a factor of $|P| = 2c$, Mikolov et al. (2018) adopt the positional weighting of Mnih and Kavukcuoglu (2013):

$$\boldsymbol{u}_{w_{t+p},p} = \boldsymbol{u}_{w_{t+p}} \odot \boldsymbol{d}_p, \tag{8}$$

Here, $\boldsymbol{d}_p \in \mathbb{R}^{D'}, p \in P$ are *positional vectors* with $D' = D$ features and $\odot : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}^D$ is the Hadamard vector product.

Compared to the subword model, the positional model more than doubles the training time, since we need to compute the Hadamard product and for each gradient update of an input vector $\boldsymbol{u}_w \in \mathbb{R}^D$, we also need to update the weights of a positional vector $\boldsymbol{d}_p \in \mathbb{R}^D$. The model can benefit from larger contexts C_t, but the training time scales linearly with the context window size c, which makes the model even more expensive.

Mikolov et al. (2018) used the positional model to improve the state-of-the-art accuracy on the English word analogy task by 5%. This demonstrates the importance of relating different positions of a sentence when creating its representation, which is also the purpose of the dense attention mechanism.

Constrained positional model The meaning of most words is partially *fixed* and partially dependent on the *narrow context* of a paragraph as well as the *broader context* that includes the conversational setting, the time and location of an utterance, and salient common ground, which may or may not be captured in the text. (Bach, 2012)

For example, consider the following sentence:

- Fruit flies like $\langle$*masked word*$\rangle$.

The sentence admits at least two interpretations:

1. what the fly likes (adj-noun-verb-$\langle$*mask*$\rangle$), or
2. how fruit flies (noun-verb-prep-$\langle$*mask*$\rangle$).

Some masked words, such as "moisture", satisfy only the first interpretation. Others, such as "a vegetable", satisfy both interpretations.

Let us now rearrange the sentence as follows:

- $\langle$*Masked word*$\rangle$ flies like fruit.

The rearranged sentence only admits the second interpretation. The masked words still include "a vegetable" but no longer "moisture".

To better adapt to these situations, context vectors $\boldsymbol{u}_{C_t}$ should contain two types of features:

1. D' narrow-context-dependent features that take the positions of context words into account and inhibit the prediction of *moisture* in the rearranged sentence, and
2. $D - D'$ fixed and broader-context-dependent features that disregard the positions of context words and encourage the prediction of "a vegetable" in both sentences.

Since CBOW does not model the broader context, we cannot distinguish between fixed and broader-context-dependent features. However, we can reduce broader-context-dependence with diachronic CBOW (Yao et al., 2018).

In the positional model, $D = D'$. Therefore, no word vector features are either fixed or broader-context-dependent. To represent the parts of a word's meaning that are fixed or dependent on the broader context, we propose to constrain the number of positionally-dependent features as follows:

$$0 < D' \ll D. \tag{9}$$

We define the constrained Hadamard vector product $\odot\colon \mathbb{R}^D \times \mathbb{R}^{D'} \to \mathbb{R}^D, D' < D$ as follows:

$$\boldsymbol{u}_{w_{t+p}} \odot \boldsymbol{d}_p = \boldsymbol{u}_{w_{t+p}} \odot [\boldsymbol{d}_p \overbrace{1 \quad \ldots \quad 1}^{D-D' \text{ times}}]. \tag{10}$$

When D' is small, the constrained positional model can reach the speed of the subword model while modeling both the fixed and the context-dependent parts of a word's meaning. The model can benefit from larger contexts C_t without making the computational complexity of training impractical, which is also the purpose of the sparse attention mechanism.

3 Experimental setup

In this section, we describe our baseline, the initialization of weights, the hyperparameter and parameter optimization, the qualitative evaluation measures, the extrinsic NLP tasks used for performance estimation, and our training corpora.

3.1 Baseline

In our experiments, we compare our constrained positional model against the subword and positional models described in Section 2.2. For the subword model, we use the implementation in Gensim 3.8.3 (Řehůřek and Sojka, 2010). Since no public implementation of the positional model exists, we release our own implementation as a free open-source software library.[1]

3.2 Initialization

For the general model, we follow the implementation of Bojanowski et al. (2017) and we initialize the features u_i of the input word vectors $\boldsymbol{u}_w$ as i.i.d. r.v.'s with continuous uniform distribution:

$$\boldsymbol{u}_w = (u_1, \ldots, u_D), u_i \sim \mathcal{U}\Big(\pm\frac{1}{D}\Big). \tag{11}$$

We initialize the output word vectors $\boldsymbol{v}_{w_t}$ to zero. For the subword model, we initialize the input subword vectors $\boldsymbol{u}_g$ as in (11) and we also initialize the output subword vectors $\boldsymbol{v}_g$ to zero.

For the positional model, Mikolov et al. (2018) do not describe the initialization of either the input subword vectors $\boldsymbol{u}_g$ or the positional vectors $\boldsymbol{d}_p$. Since no public implementation exists either, we initialize the features u_i of $\boldsymbol{u}_g$ and the features d_j of $\boldsymbol{d}_p$ as i.i.d. r.v.'s with the square-root normal distribution $\mathcal{N}^{0.5}(\mu, \sigma^2)$ of Pinelis (2018):

$$\begin{gathered}\boldsymbol{u}_g = (u_1, \ldots, u_D), \boldsymbol{d}_p = (d_1, \ldots, d_{D'}),\\ u_i \sim d_j \sim \mathcal{N}^{0.5}(\mu, \sigma^2), \mu = 0, \sigma^2 = \frac{1}{3D^2}.\end{gathered} \tag{12}$$

For technical details, see Appendix A.

For the constrained positional model, we initialize the first D' features of $\boldsymbol{u}_g$ and $\boldsymbol{d}_p$ as in (12) and the other $D - D'$ features of $\boldsymbol{u}_g$ and $\boldsymbol{d}_p$ as in (11).

[1] https://github.com/MIR-MU/pine

3.3 Optimization

In this section, we describe which hyperparameters of the subword, positional, and constrained positional models we set according to previous work and which hyperparameters we optimized using the English word analogy task. We also describe how we train the model parameters $\boldsymbol{\theta}$.

Hyperparameters For the subword, positional, and constrained positional models, we use the following hyperparameter values of Mikolov et al. (2018), which give state-of-the-art performance on the English word analogy task: We store subwords of size 3–6 in a vocabulary backed by a hash table with bucket size $2 \cdot 10^6$. We discard words with less than 5 occurrences in the corpus and we equalize the number of training samples with the low-pass threshold $r = 10^{-5}$. We use $D = 300$ features in the input and output subword vectors. For the negative sampling loss, we use $|N_{C_t}| = 10$ negative samples. For the backpropagation of the loss function L, we use the initial learning rate $\gamma_0 = 0.05$.

For the subword, positional, and constrained positional models, we optimize the context window size c, because unlike the subword model, the positional model should benefit from larger contexts. For the constrained positional model, we optimize the number of positional features D' to find the proper ratio between the fixed, narrow-context-dependent, and broader-context-dependent parts of a word's meaning. To find the optimal hyperparameter values, we maximize a model's accuracy on the English word analogy task (Mikolov et al., 2013b) using Sequential Model-Based Optimization with the Tree-structured Parzen Estimator (Bergstra et al., 2011). Like Grave et al. (2018), we restrict the vocabulary for word analogies to the $2 \cdot 10^5$ most frequent words in the training corpus.

Parameters Following Bojanowski et al. (2017), we optimize the model's parameters $\boldsymbol{\theta}$ by stochastic gradient descent over one epoch with the loss function $L(\boldsymbol{\theta})$ presented in (1) and with a linear decay of the learning rate γ_t from γ_0 to zero:

$$\gamma_t = \gamma_0 \cdot \left(1 - \frac{t}{T}\right). \tag{13}$$

We optimize the parameters in parallel using the HogWild lock-free approach of Recht et al. (2011) with 8 Intel Xeon X7560 2.26 GHz CPU cores. Since the optimization problem (1) is not sparse w.r.t. the positional vectors $\boldsymbol{d}_p$, most of which are updated at each training step, HogWild is less appropriate for the positional and constrained positional models than for the general and subword models. For all models, we report training times.

3.4 Qualitative evaluation

In this section, we propose qualitative evaluation measures, which we use to show the properties of the positional and constrained positional models. For technical details of the proposed evaluation measures, see Appendix B.

Masked word prediction For the example sentences C_t of the positional and constrained positional models from Section 2.2, we show masked words w_t in the descending order of the conditional probabilities $\Pr(w_t \mid C_t)$ from (2).

Importance of positions For each position p, we show the min-max-scaled ℓ_2-norm $\|\boldsymbol{d}_p\|$ of the positional vector $\boldsymbol{d}_p$, which measures the importance of position p for predicting masked words.

Additionally, we cluster the D' features $d_{p,j}$ of the positional vectors $\boldsymbol{d}_p$. For each cluster J and a position p, we show the mean absolute value $1/|J| \cdot \sum_{j \in J} |d_{p,j}|$, which measures the importance of position p according to cluster J.

Importance of context words For clusters J and context words w, we use the mean absolute value $1/|J| \cdot \sum_{j \in J} |u_{w,j}|$ to measure the importance of context words w using cluster J, where $u_{w,j}$ are the features of the input vector $\boldsymbol{u}_w$ for w. For each cluster J, we show context words whose importance is maximized by J.

3.5 Performance estimation

In this section, we describe the extrinsic language modeling task, which we used to estimate the performance of the input word vectors $\boldsymbol{u}_w$ produced by the subword, positional, and constrained positional models.

Language modeling For language modeling, we use a recurrent neural network (RNN) with the following architecture:

1. an input layer mapping a vocabulary V of words w to their *frozen* input vectors $\boldsymbol{u}_w$,
2. two hidden layers with $D = 300$ LSTM units,
3. a fully-connected linear layer of size $|V|$, and
4. a softmax output layer that computes a probability distribution over the vocabulary V using tied weights (Inan et al., 2017).

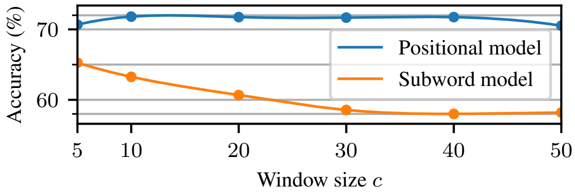

Figure 2: Word analogy accuracy of the subword and positional models trained on the 2017 English Wikipedia with different context window sizes c.

We evaluate our language model on the English datasets[2] introduced by Botha and Blunsom (2014) and we report the validation and test perplexities. We use the same preprocessing and data splits as Botha and Blunsom (2014).

To train the RNN, we use stochastic gradient descent over 50 epochs, negative log-likelihood loss, dropout 0.5, batch size 40, and an initial learning rate 20 that is divided by 4 after each epoch with no decrease of validation loss. We clip gradients with ℓ_2-norm above 0.25.

3.6 Datasets

For parameter optimization, we use the 2017 English Wikipedia[3] as the training corpus. For qualitative evaluation and performance estimation, we use the deduplicated English Common Crawl[4] as the training corpus. See the statistics in Table 1.

Dataset	Number of tokens
2017 English Wikipedia	2,423,655,228
English Common Crawl	823,575,128,431

Table 1: Our datasets and their sizes in tokens.

We preprocess the datasets by lower-casing and by tokenizing to longest sequences of Unicode characters with the *word* property (Davis and Heninger, 2020, Annex C).

4 Results

In this section, we show and discuss the results of hyperparameter and parameter optimization, qualitative evaluation, and performance estimation.

4.1 Word analogy

Table 2 shows that the positional and constrained positional models benefit from larger contexts compared to the subword model. This is further evidenced by Figure 2, which shows that the accuracy of the subword model steadily declines as the window size increases, whereas the positional model can cope with window sizes up to 40.

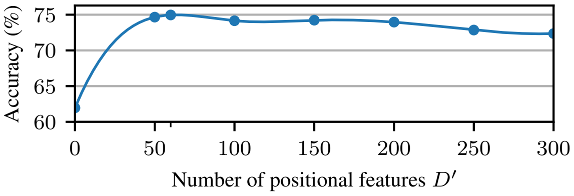

Figure 3: Word analogy accuracy of the constrained positional model trained on the 2017 English Wikipedia with different numbers of positional featured D'.

Model	c	D'	Time
Subword	5		271h 55m
Positional	15	300	970h 14m
Constrained positional	15	60	481h 30m

Table 2: The optimal context window sizes c and numbers of positional features D', and training times in hours on the English Common Crawl for the subword, positional, and constrained positional models.

Table 2 also shows that the reduction of positional dimensionality D' halves the training time of the constrained positional model compared to the positional model. Figure 3 shows that the reduction of positional dimensionality also improves the accuracy of the constrained positional model compared to the positional model.

4.2 Masked word prediction

Table 3 shows that the positional model predicts:

$$\Pr(\text{mew} \mid C_t^1) > \Pr(\text{bark} \mid C_t^1), \quad (14)$$

$$\Pr(\text{mew} \mid C_t^2) < \Pr(\text{bark} \mid C_t^2). \quad (15)$$

This matches our expectations and indicates that the model's context vectors contain narrow-context-dependent features that take the positions of context words “dogs” and “cats” into account.

Table 3 also shows that the constrained positional model predicts:

$$\Pr(\text{moisture} \mid C_t^3) > \Pr(\text{moisture} \mid C_t^4), \quad (16)$$

$$\Pr(\text{vegetable} \mid C_t^3) \approx \Pr(\text{vegetable} \mid C_t^4). \quad (17)$$

This indicates that the model contains not only narrow-context-dependent features that take the position of “moisture” into account, but also fixed and broader-context-dependent features that disregard the position of “a vegetable”.

[2] http://bothameister.github.io/

[3] https://github.com/RaRe-Technologies/gensim-data (release wiki-english-20171001)

[4] http://statmt.org/ngrams/deduped/

C_t^1 = "Unlike dogs, cats ⟨*masked word*⟩."

#	Prediction
1	cats
2	spayed
3	kennels
⋮	
1820	mew (100%)
⋮	
5581	bark (99.7%)
⋮	

C_t^2 = "Unlike cats, dogs ⟨*masked word*⟩."

#	Prediction
1	kennels
2	cats
3	puppies
⋮	
4065	bark (99.9%)
⋮	
5623	mew (99.8%)

(a) Positional model

C_t^3 = "Fruit flies like ⟨*masked word*⟩."

#	Prediction
1	fruit
2	flies
3	insects
⋮	
246	vegetable (99.9%)
⋮	
9036	moisture (69.6%)
⋮	

C_t^4 = "⟨*Masked word*⟩ flies like fruit."

#	Prediction
1	fruit
2	insects
3	flies
⋮	
259	vegetable (99.9%)
⋮	
33465	moisture (42.8%)

(b) Constrained positional model

Table 3: Masked words w_t predicted by the positional and constrained positional models for four example sentences. For selected words, we also show the conditional probability $P(w_t \mid C_t)$ in parentheses.

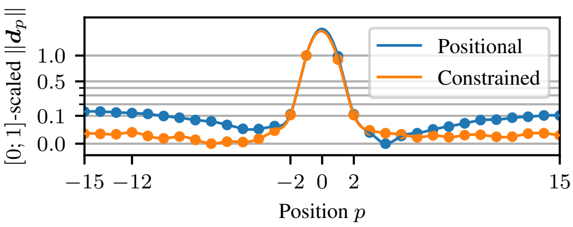

Figure 4: The importance of different positions p for predicting masked words in the positional and constrained positional models.

4.3 Positional vectors

Figure 4 shows that in the positional and constrained positional models, the importance of positions $p \in [-2; 2]$ sharply decreases with their distance from the masked word. This shows that one of the basic functions of positional weighting is the attenuation of distant context words.

In the positional model, the importance of positions $p \notin [-2; 2]$ increases with their distance from the masked word and even exceeds the importance of position $p = -2$ in the distant left context $p < -12$. In the constrained positional model, the importance of positions $p \notin [-2; 2]$ is almost constant. Below, we will explain the cause of this difference using cluster analysis.

Figure 5 shows that in the positional model, the features of positional vectors fall into three main clusters: The two smaller clusters, which we call *antepositional* and *postpositional*, and the bigger cluster, which is missing from the constrained positional model and which we call *informational*.

The antepositional and postpositional features increase the importance of positions p in anteposition $(-2, -1)$ and in postposition $(1, 2)$ of the masked word, respectively. Context words whose importance is maximized by antepositional features include "in", "for", and "coca". Context words whose importance is maximized by postpositional features include "ago", "else", and "cola". The number of antepositional and postpositional features in the positional model is 76, which is close to the $D' = 60$ positional features selected for the constrained positional model by hyperparameter optimization. This indicates that the highest task performance is reached when only antepositional and postpositional features remain.

The informational features increase the importance of positions $p \notin [-2; 2]$. Levy and Goldberg (2014) showed that context words with large input vectors have high self-information. We believe

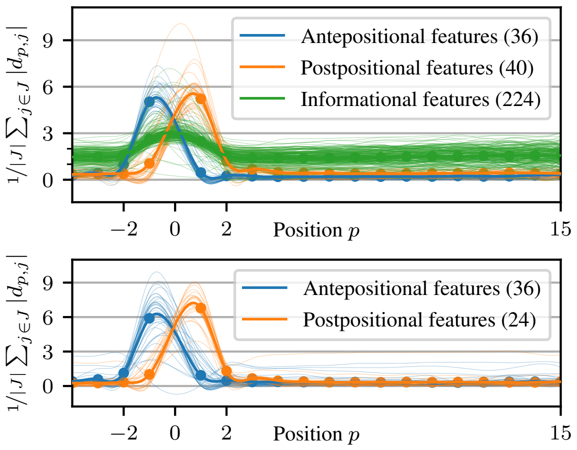

Figure 5: The importance of different positions p for predicting masked words in the positional (top) and constrained positional (bottom) models according to different clusters J of positional features. For each cluster J, we show its size $|J|$ in parentheses.

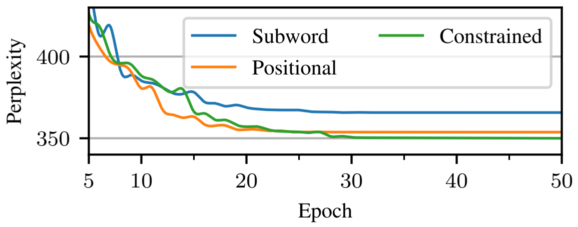

Figure 6: Validation perplexities at different epochs of RNN language models that use subword, positional, and constrained positional models as their lookup tables.

that the purpose of informational features is to amplify distant self-informational context words that indicate the general topic of a sentence. Context words whose importance is maximized by informational features include "finance", "sports", and "politics". In the constrained positional model, the informational features $d_{p,j}$ are effectively replaced by ones, which is close to what the positional model has learnt.

4.4 Language modeling

Figure 6 shows that the positional and constrained positional models consistently outperform the subword model during the training of RNN language models. Figure 6 also shows that RNN language models have converged and that training for more epochs would not have improved their perplexity.

Table 4 shows that the constrained positional model produces word vectors that are better suited for initializing the lookup tables of RNN language models than the subword and positional models.

	Subword	Position.	Cons.
Test perplexity	360.91	347.52	**343.13**

Table 4: Test perplexities of RNN language models that use subword, positional, and constrained positional models as lookup tables. Best result is **emphasized**.

5 Conclusion

In our work, we have related the attention mechanism to the positional model of Mikolov et al. (2018) and we have adapted the attention sparsification techniques of Zaheer et al. (2020) to develop our constrained positional model, which is more expressive, better-suited to language modeling, equally interpretable, and practically fast.

Future work should focus on quantitative evaluation of the constrained positional model on additional extrinsic NLP tasks, both alone and in tandem with deep attention-based language models.

References

Kent Bach. 2012. Context dependence (such as it is). *The Continuum Companion to the Philosophy of Language*, pages 153–184.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2016. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473v7*.

Iz Beltagy, Matthew E Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150v2*.

James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24:2546–2554.

Piotr Bojanowski et al. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Jan A. Botha and Phil Blunsom. 2014. Compositional morphology for word representations and language modelling. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML'14, page II–1899–II–1907. JMLR.org.

Delphine Charlet and Geraldine Damnati. 2017. Simbow at SemEval-2017 task 3: Soft-cosine semantic similarity between questions for community question answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 315–319.

Xinxiong Chen, Zhiyuan Liu, and Maosong Sun. 2014. A unified model for word sense representation and disambiguation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1025–1035.

Jianpeng Cheng, Li Dong, and Mirella Lapata. 2016. Long short-term memory-networks for machine reading. *arXiv preprint arXiv:1601.06733v7*.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509v1*.

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014a. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259v2*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014b. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078v3*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D Manning. 2019. What does BERT look at? an analysis of BERT's attention. *arXiv preprint arXiv:1906.04341v1*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Mark Davis and Andy Heninger. 2020. Unicode technical standard #18: Unicode regular expressions. *The Unicode Consortium*. Version 21.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805v2*.

Edouard Grave, Piotr Bojanowski, Prakhar Gupta, Armand Joulin, and Tomáš Mikolov. 2018. Learning word vectors for 157 languages. *arXiv preprint arXiv:1802.06893v2*.

Michael U. Gutmann and Aapo Hyvärinen. 2012. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *The Journal of Machine Learning Research, JLMR*, 13(1):307–361.

Hakan Inan, Khashayar Khosravi, and Richard Socher. 2017. Tying word vectors and word classifiers: A loss framework for language modeling. *arXiv preprint arXiv:1611.01462v3*.

Daniel Kahneman. 2011. *Thinking, Fast and Slow*. Farrar, Straus and Giroux, New York.

Matt Kusner, Yu Sun, Nicholas Kolkin, and Kilian Weinberger. 2015. From Word Embeddings To Document Distances. In *International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 957–966, Lille, France. PMLR.

Omer Levy and Yoav Goldberg. 2014. Neural word embedding as implicit matrix factorization. *Advances in Neural Information Processing Systems*, 27:2177–2185.

Wentian Li. 1992. Random texts exhibit zipf's-law-like word frequency distribution. *IEEE Transactions on information theory*, 38(6):1842–1845.

Tomáš Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781v3*.

Tomáš Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Tomáš Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119. Curran Associates, Inc.

Andriy Mnih and Koray Kavukcuoglu. 2013. Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in Neural Information Processing Systems*, volume 26, pages 2265–2273. Curran Associates, Inc.

Vít Novotný, Petr Sojka, Michal Štefánik, and Dávid Lupták. 2020. Three is better than one. In *CEUR Workshop Proceedings*, page 30, Thessaloniki, Greece.

Ellen Peters, Daniel Västfjäll, Paul Slovic, CK Mertz, Ketti Mazzocco, and Stephan Dickert. 2006. Numeracy and decision making. *Psychological science*, 17(5):407–413.

Iosif Pinelis. 2018. The exp-normal distribution is infinitely divisible. *arXiv preprint arXiv:1803.09838v1*.

S. K. Ravshan. 2018. Factor analysis and uniform distributions. Visited on 2020-12-05.

Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. 2011. Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in Neural Information Processing Systems*, volume 24, pages 693–701. Curran Associates, Inc.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27:3104–3112.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

David E. Wright. 2008. Do Words Have Inherent Meaning? *ETC: A Review of General Semantics*, 65(2):177–190.

Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. 2018. Dynamic word embeddings for evolving semantic discovery. In *Proceedings of the eleventh acm international conference on web search and data mining*, pages 673–681.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062v1*.

George Kinsley Zipf. 1932. *Selective Studies and the Principle of Relative Frequency in Language*. Harvard University Press, Cambridge MA.

Radim Řehůřek and Petr Sojka. 2010. Software framework for topic modelling with large corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, pages 45–50, Valletta, Malta. ELRA.

A Initialization of the positional model

In this appendix, we expand on Section 3.2 by describing three initialization options for the positional model. We discuss the properties of the initialization options and their practical effect on the training of the positional model.

A.1 Positions same as vanilla subwords

The simplest option is to initialize both the input subword vectors $\boldsymbol{u}_g$ and the positional vectors $\boldsymbol{d}_p$ as in (11). In practice, this decreases the variance of the context vector $\boldsymbol{u}_{C_t}$ in the positional model compared to the subword model:

$$\begin{aligned}
&\mathbb{V}\mathrm{ar}\Bigg[\frac{1}{|C_t|}\sum_{\substack{w\in C_t\\ g\in G_w}} \boldsymbol{u}_g \odot \boldsymbol{d}_p\Bigg] \qquad (18)\\
&= \frac{1}{|C_t|^2}\sum_{\substack{w\in C_t\\ g\in G_w}} \Big(\mathbb{V}\mathrm{ar}[\boldsymbol{u}_g \odot \boldsymbol{d}_p] = \mathbb{E}\big[\boldsymbol{u}_g^2\big] \odot \mathbb{E}\big[\boldsymbol{d}_p^2\big]\Big)\\
&= \frac{1}{|C_t|^2}\sum_{\substack{w\in C_t\\ g\in G_w}} \frac{\mathbf{1}}{\mathbf{9}\boldsymbol{D}^{\mathbf{4}}} \ll \frac{1}{|C_t|^2}\sum_{\substack{w\in C_t\\ g\in G_w}} \frac{\mathbf{1}}{\mathbf{3}\boldsymbol{D}^{\mathbf{2}}}\\
&= \frac{1}{|C_t|^2}\sum_{\substack{w\in C_t\\ g\in G_w}} \mathbb{V}\mathrm{ar}[\boldsymbol{u}_g] = \mathbb{V}\mathrm{ar}\Bigg[\frac{1}{|C_t|}\sum_{\substack{w\in C_t\\ g\in G_w}} \boldsymbol{u}_g\Bigg].
\end{aligned}$$

See also Figure 7. The decrease in $\mathbb{V}\mathrm{ar}[\boldsymbol{u}_{C_t}]$ decreases $\mathbb{V}\mathrm{ar}[\nabla L]$ and therefore it also decreases the effective learning rate of the positional model. As D increases, the context vector $\boldsymbol{u}_{C_t}$ quickly tends to zero due to $\mathbb{V}\mathrm{ar}[\boldsymbol{u}_g \odot \boldsymbol{d}_p] = {}^{\mathbf{1}}\!/_{\mathbf{9}\boldsymbol{D}^{\mathbf{4}}}$.

A.2 Identity positions and vanilla subwords

To keep the effective learning rate of the positional model the same as in the subword model, it is sufficient to keep the distribution of the context vector $\boldsymbol{u}_{C_t}$ the same as in the subword model. To achieve this, we initialize the input subword vectors $\boldsymbol{u}_g$ as in (11) and the positional vectors $\boldsymbol{d}_p$ to one. Intuitively, the training starts with no positional weighting and the positional vectors are learnt later. In practice, $\boldsymbol{d}_p \gg \boldsymbol{u}_g$, causing the gradient $\nabla_{\boldsymbol{u}_g} L$ to explode for $D > 600$ soon after the training has begun. This leads to numerical instability and the model parameters $\boldsymbol{\theta}$ tend to NaN as the training continues.

A.3 Positions same as subwords

To keep the effective learning rate of the positional model the same as in the subword model and to avoid exploding gradients, it is sufficient to keep

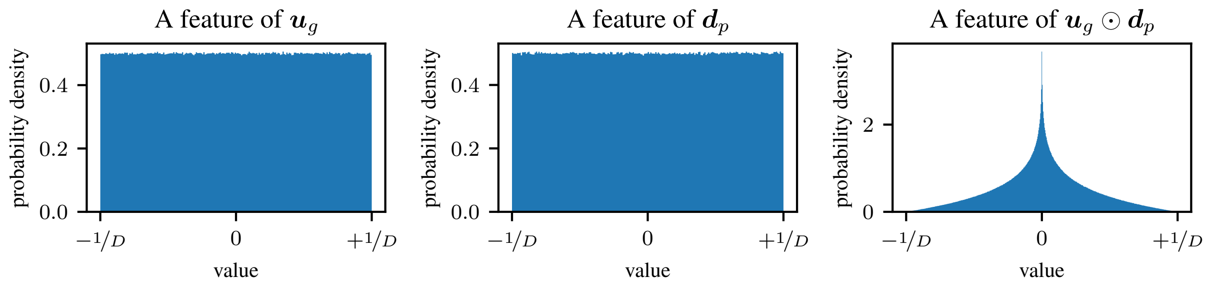

Figure 7: Probability densities of feature values in the input subword vectors $\boldsymbol{u}_g$, the positional vectors $\boldsymbol{d}_p$, and their product $\boldsymbol{u}_g \odot \boldsymbol{d}_p$ with the *positions same as vanilla subwords* initialization to $\mathcal{U}(\pm 1/D), D = 1$. Since $\mathbb{V}\mathrm{ar}[\boldsymbol{u}_g] \gg \mathbb{V}\mathrm{ar}[\boldsymbol{u}_g \odot \boldsymbol{d}_p]$, the effective learning rate of the positional model is smaller than in the subword model.

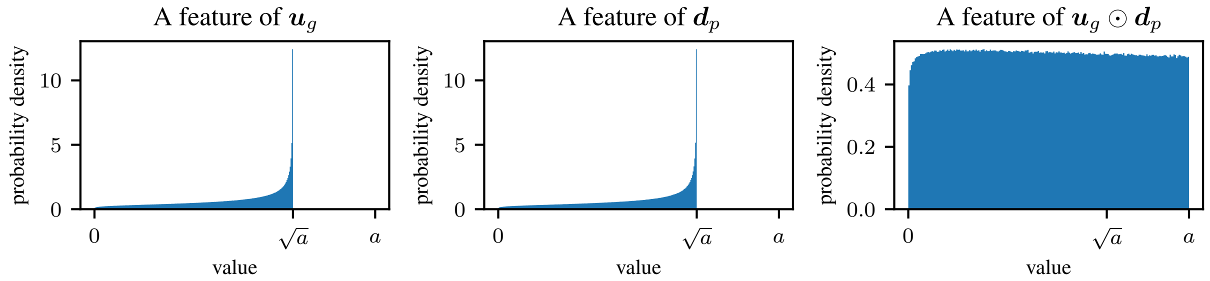

Figure 8: Probability densities of feature values in the input subword vectors $\boldsymbol{u}_g$, the positional vectors $\boldsymbol{d}_p$, and their product $\boldsymbol{u}_g \odot \boldsymbol{d}_p$ with the initialization to $\mathcal{U}^{0.5}(0, a), a = 2$. For our use, we would need $\mathcal{U}^{0.5}(\pm 1/D)$ instead.

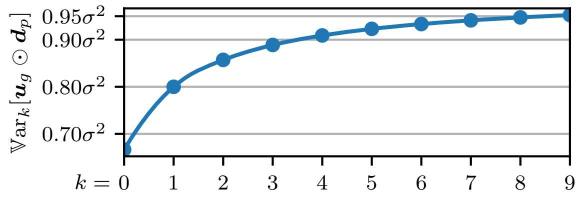

Figure 9: The variances $\mathbb{V}\mathrm{ar}_k[\boldsymbol{u}_g \odot \boldsymbol{d}_p]$ for the feature values in the product $\boldsymbol{u}_g \odot \boldsymbol{d}_p$ of the input subword vectors $\boldsymbol{u}_g$ and the positional vectors $\boldsymbol{d}_p$ with the initialization to the square-root normal distribution $\mathcal{N}^{0.5}(0, \sigma^2)$, when we approximate the infinite sequence $(a_n)_{n=0}^{\infty}$ in the definition of $\mathcal{N}^{0.5}$ by first $k+1$ elements $(a_n)_{n=0}^{k}$.

the distribution of the context vector $\boldsymbol{u}_{C_t}$ the same as in the subword model and to initialize both the input subword vectors $\boldsymbol{u}_g$ and the positional vectors $\boldsymbol{d}_p$ from the same distribution. We could achieve this by initializing the features u_i of $\boldsymbol{u}_g$ and the features d_j of $\boldsymbol{d}_p$ as i.i.d. r.v.'s with the square-root distribution $\mathcal{U}^{0.5}(\pm 1/D)$ such that $u_i \cdot d_j \sim \mathcal{U}(\pm 1/D)$. Although an approximation of $\mathcal{U}^{0.5}(0, a)$ using the β-distribution is known (Ravshan, 2018), see Figure 8, it does not extend to $\mathcal{U}^{0.5}(\pm 1/D)$, so we need another approach.

Assuming the context C_t is sufficiently large, then by the central limit theorem, the features of the context vector $\boldsymbol{u}_{C_t}$ in the subword model have the normal distribution $\mathcal{N}(\mu, \sigma^2/|C_t|)$, where $\mu = \mathbb{E}[\mathcal{U}(\pm 1/D)] = 0, \sigma^2 = \mathbb{V}\mathrm{ar}[\mathcal{U}(\pm 1/D)] = 1/3D^2$. To achieve the same distribution with the positional model, we initialize the features u_i of $\boldsymbol{u}_g$ and the features d_j of $\boldsymbol{d}_p$ as i.i.d. r.v.'s with some continuous distribution $\mathcal{X}$ such that $\mathbb{E}[u_i \cdot d_j] = \mu$ and $\mathbb{V}\mathrm{ar}[u_i \cdot d_j] = \sigma^2$. In our initialization, we use as $\mathcal{X}$ the square-root normal distribution $\mathcal{N}^{0.5}(\mu, \sigma^2)$ of Pinelis (2018), see Figure 10. The continuous uniform $\mathcal{U}(\pm \sqrt[4]{3}/\sqrt{D})$ is also an option.

The definition of $\mathcal{N}^{0.5}$ by Pinelis (2018) contains a sum of an infinite sequence $(a_n)_{n=0}^{\infty}$:

$$\begin{aligned} \mathcal{N}^{0.5}(\mu, \sigma^2) &= \epsilon \cdot e^{\sum_{n=0}^{\infty} a_n} \cdot \sqrt{\sigma} + \sqrt{\mu}, \\ a_n &= \frac{1}{4} \cdot \ln\left(1 + \frac{1}{\max(1, n)}\right) - \frac{G_n}{2n+1}, \end{aligned} \tag{19}$$

where $(G_n)_{n=0}^{\infty}$ are i.i.d. r.v.'s with $\mathrm{Gamma}(1/2, 1)$ distribution and ϵ is a Rademacher r.v. independent of all G_n. Since $\lim_{n\to\infty} a_n = 0$, we can approximate $\sum_{n=0}^{\infty} a_n$ by its first $k+1$ elements, but we need guarantees about $\mathbb{E}[u_i \cdot d_j]$ and $\mathbb{V}\mathrm{ar}[u_i \cdot d_j]$. We can see that $\mathbb{E}[u_i \cdot d_j] = \mu$ for any k:

$$\begin{aligned} &\mathbb{E}[u_i] = \mathbb{E}[d_j] \\ &= \mathbb{E}[\epsilon] \cdot \mathbb{E}\left[e^{\sum_{n=0}^{k} a_n}\right] \cdot \mathbb{E}[\sqrt{\sigma}] + \mathbb{E}[\sqrt{\mu}] \\ &= 0 \cdot \mathbb{E}\left[e^{\sum_{n=0}^{k} a_n}\right] \cdot \mathbb{E}[\sqrt{\sigma}] + \sqrt{\mu} = \sqrt{\mu}, \\ &\mathbb{E}[u_i \cdot d_j] = \mathbb{E}[u_i] \cdot \mathbb{E}[d_j] = (\sqrt{\mu})^2 = \mu. \end{aligned} \tag{20}$$

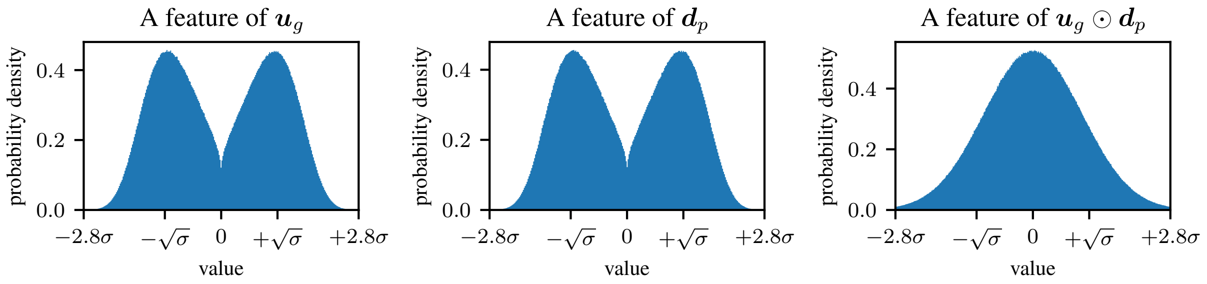

Figure 10: Probability densities of feature values in the input subword vectors $\boldsymbol{u}_g$, the positional vectors $\boldsymbol{d}_p$, and their product $\boldsymbol{u}_g \odot \boldsymbol{d}_p$ with the *positions same as subwords* initialization to the square-root normal distribution $\mathcal{N}^{0.5}(0, \sigma^2)$, where $\sigma^2 = {}^1\!/_{3D^2}$ and $D = 1$.

We will denote the variance for a given k as $\mathbb{V}\mathrm{ar}_k$:

$$\begin{aligned}\mathbb{V}\mathrm{ar}_k[u_i] &= \mathbb{V}\mathrm{ar}_k[d_j]\\ &= \mathbb{V}\mathrm{ar}\big[\epsilon \cdot e^{\sum_{n=0}^{k} a_n} \cdot \sqrt{\sigma} + \sqrt{\mu}\big]\\ &= \Big(\prod_{n=0}^{k} \mathbb{E}[e^{a_n}]^2 + \mathbb{V}\mathrm{ar}\Big[\prod_{n=0}^{k} e^{a_n}\Big]\Big) \cdot \sigma,\\ \mathbb{V}\mathrm{ar}_k[u_i \cdot d_j] &= 2\mu \cdot \mathbb{V}\mathrm{ar}_k[u_i] + \mathbb{V}\mathrm{ar}_k[u_i]^2.\end{aligned} \tag{21}$$

In our initialization, we approximate $\sum_{n=0}^{\infty} a_n$ by $\sum_{n=0}^{k} a_n, k = 9$. As we show in Figure 9, this guarantees $\mathbb{E}[u_i \cdot d_j] = \mu, {}^{\mathbb{V}\mathrm{ar}[u_i \cdot d_j]}\!/_{\sigma^2} \in (0.95; 1]$.

B Qualitative evaluation measures

In this appendix, we expand on Section 3.4 by showing how the proposed qualitative evaluation measures relate to the conditional probability $\Pr(w_t \mid C_t)$ from (2). For a fixed set of negative samples N_{C_t}, $\Pr(w_t \mid C_t)$ is a strictly increasing transformation of the scoring function $s(w_t, C_t)$ from (3). Without loss of generality, our proofs will focus on $s(w_t, C_t)$ rather than on $\Pr(w_t \mid C_t)$.

B.1 Importance of positions

All else being constant, the ℓ_2-norm $\|\boldsymbol{d}_p\|$ is an asymptotic upper bound on $|s(w_t, C_t)|$:

$$\begin{aligned}|s(w_t, C_t)| &= |\boldsymbol{u}_{C_t}^{\mathsf{T}} \cdot \boldsymbol{v}_{w_t}|\\ &= \frac{1}{|P|} \cdot |(\boldsymbol{u}_{w_{t+p}} \odot \boldsymbol{d}_p)^{\mathsf{T}} \boldsymbol{v}_{w_t} + \dots|\\ &\leq \frac{1}{|P|} \cdot (\|\boldsymbol{u}_{w_{t+p}}\| \cdot \|\boldsymbol{d}_p\| \cdot \|\boldsymbol{v}_{w_t}\| + \|\dots\|),\\ |s(w_t, C_t)| &\in \mathcal{O}(\|\boldsymbol{d}_p\|).\end{aligned} \tag{22}$$

All else being constant, $\sum_{j \in J} |d_{p,j}|$ is also an asymptotic upper bound on $|s(w_t, C_t)|$:

$$\begin{aligned}|s(w_t, C_t)| &= |\boldsymbol{u}_{C_t}^{\mathsf{T}} \cdot \boldsymbol{v}_{w_t}|\\ &= \frac{1}{|P|} \cdot \Big|\sum_{j \in J} u_{w_{t+p},j} \cdot d_{p,j} \cdot v_{w_t,j} + \dots\Big|\\ &\leq \frac{1}{|P|} \cdot \Big(\sum_{j \in J} |u_{w_{t+p},j}| \cdot |d_{p,j}| \cdot |v_{w_t,j}| + |\dots|\Big),\\ |s(w_t, C_t)| &\in \mathcal{O}\Big(\sum_{j \in J} |d_{p,j}|\Big),\end{aligned} \tag{23}$$

where $u_{w_{t+p},j}$ are features of the input vector $\boldsymbol{u}_{w_{t+p}}$ for the context word w_{t+p} at position p and $v_{w_t,j}$ are features of the output vector $\boldsymbol{v}_{w_t}$ for the masked word w_t.

B.2 Importance of context words

All else being constant, $\sum_{j \in J} |u_{w,j}|$ is an asymptotic upper bound on the expected absolute difference $\mathbb{E}|s(w_t, C_t) - s(w_t, C_t')|$ for a fixed context word w and random-valued masked words w_t and contexts C_t, C_t', where w is at position p_1 in C_t and at position p_2 in C_t':

$$\begin{aligned}&\mathbb{E}|s(w_t, C_t) - s(w_t, C_t')|\\ &= \frac{1}{|P|} \cdot \sum_{w_t, C_t, C_t'} \Big|\sum_{j \in J} u_{w,j} \cdot (d_{p_1,j} - d_{p_2,j}) \cdot v_{w_t,j}\\ &\qquad + \dots\Big| \cdot \Pr(w_t, C_t, C_t')\\ &\leq \frac{1}{|P|} \cdot \sum_{\substack{w_t, C_t, C_t'\\ j \in J}} |u_{w,j}| \cdot |(d_{p_1,j} - d_{p_2,j}) \cdot v_{w_t,j}|\\ &\qquad \cdot \Pr(w_t, C_t, C_t') + |\dots| \cdot \Pr(w_t, C_t, C_t'),\\ &\mathbb{E}|s(w_t, C_t) - s(w_t, C_t')| \in \mathcal{O}\Big(\sum_{j \in J} |u_{w,j}|\Big).\end{aligned} \tag{24}$$